Tingimuslause, tsükkel

Tingimuslause

Tihti on meil vaja kontrollida koodi kulgemist, selleks on mitmeid võimalusi. If-lause:

```
x=5
if (x<10) {
  print("x on väikem kui 10")
}
```

```
## [1] "x on väikem kui 10"
```

If-lause koos else-ga:

```
if (x<5) {
  print("x on suurem kui 5")
} else {
  print("x on suurem või võrdne 5-ga")
}
```

```
## [1] "x on suurem või võrdne 5-ga"
```

Lisaks von olemas ka if else:

```
x <- 0
if (x < 0) {
print("Negative number")
} else if (x > 0) {
print("Positive number")
} else {
print("Zero")
}
```

```
## [1] "Zero"
```

Tingimused võivad olla keerulisemad:

```
x=5
y=10
nimi='Juhan'

if (x<10 & y>=10 & nimi=='Juhan') {
  print('If püüdis kinni')
}
```

```
## [1] "If püüdis kinni"
```

```
if (x<10 & y>10 & nimi=='Juhan') {
  print('If püüdis kinni')
} else {
  print('If ei püüdnud kinni')
}
```

```
## [1] "If ei püüdnud kinni"
```

```
#ekstra sulud on vaja, kuna tahan kogu loogilise avaldise tulemuse muuta vastupidiseks
if (!(x<10 & y>10 & nimi=='Juhan')) {
```

```
  print('If püüdis kinni')
} else {
  print('If ei püüdnud kinni')
}
```

```
## [1] "If püüdis kinni"
```

```
if (x<10 | y>10 & nimi=='Juhan') {
  print('If püüdis kinni')
} else {
  print('If ei püüdnud kinni')
}
```

```
## [1] "If püüdis kinni"
```

Tsükkel

Lihtsaimad tsükli haldamise vahendid on for ja while tsükkel. Neid tegelikult väga palju R-si ei kasutata.

```
for (year in 2010:2015){
  print(paste("The year is", year))
}
```

```
## [1] "The year is 2010"
## [1] "The year is 2011"
## [1] "The year is 2012"
## [1] "The year is 2013"
## [1] "The year is 2014"
## [1] "The year is 2015"
```

```
for (year in c(2010,2011,2012,2013,2014,2015)){
  print(paste("The year is", year))
}
```

```
## [1] "The year is 2010"
## [1] "The year is 2011"
## [1] "The year is 2012"
## [1] "The year is 2013"
## [1] "The year is 2014"
## [1] "The year is 2015"
```

```
#mida see koodijupp teeb?
for (i in 1:10) {
  if (!i %% 2){
    next #viib koodi järgmisesse tsükli ringi
  }
  print(i)
}
```

```
## [1] 1
## [1] 3
## [1] 5
## [1] 7
## [1] 9
```

While tsükkel. While tsükli juures on oluline tagada, et ei tekiks lõpmatut tsüklit, muidu R jookseb lihtsalt kokku.

```
i <- 1
while (i < 6) {
  print(i)
  i = i+1 #kui see välja kommenteerida, tekib lõpmatu tsükkel
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
```

Repeat tsükkel. Siingi tuleb olla ettevaatlik, et ei tekiks lõpmatut tsüklit.

```
x <- 1
repeat {
  print(x)
  x = x+1
  if (x == 6){
    break #lõpetab tsükli
    }
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
```

Tsükleid saab üksteise sisse panna.

```
# Create a 30 x 30 matrix (of 30 rows and 30 columns)
mymat <- matrix(nrow=30, ncol=30)

for(i in 1:dim(mymat)[1]) {#võtab maatriks read ükshaaval
  for(j in 1:dim(mymat)[2]) { #võtab konkreetse rea veerud ükshaaval
    mymat[i,j] = i*j #väärtustab maatriksi konkreetse elemendi
  }
}

# Just show the upper left 10x10 chunk
mymat[1:10, 1:10]
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]    1    2    3    4    5    6    7    8    9    10
##  [2,]    2    4    6    8   10   12   14   16   18    20
##  [3,]    3    6    9   12   15   18   21   24   27    30
##  [4,]    4    8   12   16   20   24   28   32   36    40
##  [5,]    5   10   15   20   25   30   35   40   45    50
##  [6,]    6   12   18   24   30   36   42   48   54    60
##  [7,]    7   14   21   28   35   42   49   56   63    70
##  [8,]    8   16   24   32   40   48   56   64   72    80
##  [9,]    9   18   27   36   45   54   63   72   81    90
## [10,]   10   20   30   40   50   60   70   80   90   100
```

Vektoriseerimine

Tsükliga:

```
#halb näide, aeglane
algusaeg <- proc.time()

#seab juhuslikkuse
set.seed(42)

m=100
n=100

# Create matrix of normal random numbers
mymat <- replicate(m, rnorm(n))

# Transform into data frame
mydframe <- data.frame(mymat)

for (i in 1:m) {
  for (j in 1:n) {
    mydframe[i,j]<-mydframe[i,j] + 10*sin(0.75*pi)
    #print(mydframe)
  }
}
print('kulunud aeg')
```

```
## [1] "kulunud aeg"
```

```
print(proc.time()-algusaeg)
```

```
##    user  system elapsed
##    0.41    0.00    0.41
```

```
mydframe[1:10,1:10]
```

```
##          X1       X2       X3       X4       X5       X6       X7       X8
## 1  8.442026 8.272033 5.070139 7.066447 8.405980 8.100209 6.822585 7.365760
## 2  6.506370 8.115819 7.404845 7.831310 6.201796 7.985843 7.493388 7.463809
## 3  7.434196 6.067859 8.242393 7.110059 7.126555 7.068612 8.058721 6.070224
## 4  7.703930 8.919550 9.130607 7.806140 7.120135 7.207077 7.906636 6.745341
## 5  7.475336 6.404294 5.694206 6.924595 6.492712 6.350914 6.410546 6.062719
## 6  6.964943 7.176582 5.920212 7.013180 6.072329 6.872943 8.635137 6.435636
## 7  8.582590 6.648812 6.365246 7.553437 7.068635 6.041859 5.448092 5.861227
## 8  6.976409 6.948718 6.017012 8.064011 7.726580 6.104112 7.934964 5.954604
## 9  9.089492 7.259261 6.425324 5.824672 8.547910 5.850255 6.559465 7.700949
## 10 7.008354 7.190229 6.885690 7.037580 5.161915 7.907276 5.153703 6.798546
##          X9      X10
## 1  7.759876 8.012992
## 2  7.796151 6.822454
## 3  7.288448 7.167547
## 4  6.869411 6.637137
## 5  5.705378 9.249736
## 6  6.762130 4.112288
## 7  6.618165 7.151956
## 8  7.734297 7.181206
```

```
## 9  8.379697 7.284516
## 10 7.572108 5.513248
```

Vektoriseeritud:

```
algusaeg <- proc.time()
set.seed(42)
m=100
n=100
mymat <- replicate(m, rnorm(n))
mydframe <- data.frame(mymat)
mydframe <- mydframe + 10*sin(0.75*pi)
print('kulunud aeg')
```

```
## [1] "kulunud aeg"
```

```
proc.time()-algusaeg
```

```
##    user  system elapsed
##    0.02    0.00    0.02
```

```
mydframe[1:10,1:10]
```

```
##          X1       X2       X3       X4       X5       X6       X7       X8
## 1  8.442026 8.272033 5.070139 7.066447 8.405980 8.100209 6.822585 7.365760
## 2  6.506370 8.115819 7.404845 7.831310 6.201796 7.985843 7.493388 7.463809
## 3  7.434196 6.067859 8.242393 7.110059 7.126555 7.068612 8.058721 6.070224
## 4  7.703930 8.919550 9.130607 7.806140 7.120135 7.207077 7.906636 6.745341
## 5  7.475336 6.404294 5.694206 6.924595 6.492712 6.350914 6.410546 6.062719
## 6  6.964943 7.176582 5.920212 7.013180 6.072329 6.872943 8.635137 6.435636
## 7  8.582590 6.648812 6.365246 7.553437 7.068635 6.041859 5.448092 5.861227
## 8  6.976409 6.948718 6.017012 8.064011 7.726580 6.104112 7.934964 5.954604
## 9  9.089492 7.259261 6.425324 5.824672 8.547910 5.850255 6.559465 7.700949
## 10 7.008354 7.190229 6.885690 7.037580 5.161915 7.907276 5.153703 6.798546
##          X9      X10
## 1  7.759876 8.012992
## 2  7.796151 6.822454
## 3  7.288448 7.167547
## 4  6.869411 6.637137
## 5  5.705378 9.249736
## 6  6.762130 4.112288
## 7  6.618165 7.151956
## 8  7.734297 7.181206
## 9  8.379697 7.284516
## 10 7.572108 5.513248
```

Vahepõikena, mida aeg näitab: "the **user time** relates to the execution of the code, the **system time** relates to system processes such as opening and closing files, and the **elapsed time** is the difference in times since you started the stopwatch (and will be equal to the sum of user and system times if the chunk of code was run altogether and single-threaded). Note that the elapsed time may be shorter than the sum of the user time and system time if multiple threads were used to execute the expression" (https://stats.idre.ucla.edu/r/faq/how-can-i-time-my-code/)

Me oleme tegelikult vektoriseeritud kuju näinud:

```
vec1=c(1,2,3)
vec2=c(10,20, 30)
vec1+vec2
```

```
## [1] 11 22 33
```